

**What Is Claimed Is:**

- 1        1. A method for dynamically adjusting the aggressiveness of an  
2 execute-ahead processor, comprising:
  - 3            executing instructions in an execute-ahead mode, wherein instructions that  
4 cannot be executed because of an unresolved data dependency are deferred, and  
5 other non-deferred instructions are executed in program order, and wherein if a  
6 non-data-dependent stall condition is encountered, the execute-ahead processor  
7 enters a scout mode, wherein instructions are speculatively executed to prefetch  
8 future loads, but results are not committed to the architectural state of the execute-  
9 ahead processor;
  - 10            if an unresolved data dependency is resolved during the execute-ahead  
11 mode, executing deferred instructions in a deferred mode;
    - 12            wherein if some instructions are deferred again during the deferred mode,  
13 the method further comprises,
      - 14            determining whether to resume execution in the execute-  
15 ahead mode,
      - 16            if it is determined to do so, resuming execution in the  
17 execute-ahead mode, and
      - 18            otherwise resuming execution in a non-aggressive mode.
- 1        2. The method of claim 1, wherein resuming execution in the non-  
2 aggressive execution mode involves remaining in the deferred mode until all  
3 deferred instructions are executed and the execute-ahead processor returns to a  
4 normal execution mode.

1           3.       The method of claim 1, wherein resuming execution in the non-  
2 aggressive mode involves resuming execution in a non-aggressive execute-ahead  
3 mode, wherein if a non-data-dependent stall condition is encountered, the execute-  
4 ahead processor does not enter the scout mode, but instead waits for the non-data-  
5 dependent stall condition to be resolved, or for an unresolved data dependency to  
6 return, before proceeding.

1           4.       The method of claim 1, wherein prior to executing instructions in  
2 execute-ahead mode, the method further comprises entering the execute-ahead  
3 mode by:

4              issuing instructions for execution in program order during a normal  
5 execution mode;

6              upon encountering an unresolved data dependency during execution of an  
7 instruction,

8                  generating a checkpoint that can subsequently be used to  
9                  return execution to the point of the instruction, and  
10                 executing subsequent instructions in the execute-ahead  
11                 mode.

1           5.       The method of claim 4, wherein if the launch point stall condition  
2 (the unresolved data dependency or the non-data-dependent stall condition that  
3 originally caused the execute-ahead processor to exit the normal execution mode)  
4 is finally resolved, the method further comprises using the checkpoint to resume  
5 execution in the normal execution mode from the launch point instruction (the  
6 instruction that originally encountered the launch point stall condition).

1           6.       The method of claim 1, wherein executing deferred instructions in  
2 the deferred mode involves:  
3              issuing deferred instructions for execution in program order;  
4              deferring execution of deferred instructions that still cannot be executed  
5 because of unresolved data dependencies; and  
6              executing other deferred instructions that are able to be executed in  
7 program order.

1           7.       The method of claim 6, wherein if all deferred instructions are  
2 executed in the deferred mode, the method further comprises returning to a  
3 normal execution mode to resume normal program execution from the point  
4 where the execute-ahead mode left off.

1           8.       The method of claim 1, wherein the unresolved data dependency  
2 can include:  
3              a use of an operand that has not returned from a preceding load miss;  
4              a use of an operand that has not returned from a preceding translation  
5 lookaside buffer (TLB) miss;  
6              a use of an operand that has not returned from a preceding full or partial  
7 read-after-write (RAW) from store buffer operation; and  
8              a use of an operand that depends on another operand that is subject to an  
9 unresolved data dependency.

1           9.       The method of claim 1, wherein the non-data-dependent stall  
2 condition can include:  
3              a memory barrier operation;  
4              a load buffer full condition; and

5 a store buffer full condition.

1 10. An apparatus that dynamically adjusts the aggressiveness of an  
2 execute-ahead processor, comprising:  
3 an execution mechanism configured to execute instructions in an execute-  
4 ahead mode, wherein instructions that cannot be executed because of an  
5 unresolved data dependency are deferred, and other non-deferred instructions are  
6 executed in program order, and wherein if a non-data-dependent stall condition is  
7 encountered, the execution mechanism is configured to enter a scout mode,  
8 wherein instructions are speculatively executed to prefetch future loads, but  
9 results are not committed to the architectural state of the execute-ahead processor;  
10 wherein if an unresolved data dependency is resolved during the execute-  
11 ahead mode, the execution mechanism is configured to execute deferred  
12 instructions in a deferred mode;  
13 wherein if some instructions are deferred again during the deferred mode,  
14 the execution mechanism is configured to,  
15 determine whether to resume execution in the execute-  
16 ahead mode,  
17 if it is determined to do so, to resume execution in the  
18 execute-ahead mode, and  
19 otherwise to resume execution in a non-aggressive mode.

1 11. The apparatus of claim 10, wherein while resuming execution in  
2 the non-aggressive execution mode, the execution mechanism is configured to  
3 remain in the deferred mode until all deferred instructions are executed and the  
4 execution mechanism returns to a normal execution mode.

1           12. The apparatus of claim 10, wherein while resuming execution in  
2 the non-aggressive execution mode, the execution mechanism is configured to  
3 resume execution in a non-aggressive execute-ahead mode, wherein if a non-data-  
4 dependent stall condition is encountered, the execution mechanism does not enter  
5 the scout mode, but instead waits for the non-data-dependent stall condition to be  
6 resolved, or for an unresolved data dependency to return, before proceeding.

1           13. The apparatus of claim 10, wherein prior to executing instructions  
2 in execute-ahead mode, the execution mechanism is configured to enter the  
3 execute-ahead mode by:

4           issuing instructions for execution in program order during a normal  
5 execution mode;

6           upon encountering an unresolved data dependency during execution of an  
7 instruction,

8           generating a checkpoint that can subsequently be used to  
9           return execution at to the point of the instruction, and  
10           executing subsequent instructions in the execute-ahead  
11           mode.

1           14. The apparatus of claim 13, wherein if the launch point stall  
2 condition (the unresolved data dependency or the non-data-dependent stall  
3 condition that originally caused the execution mechanism to exit the normal  
4 execution mode) is finally resolved, the execution mechanism is configured to use  
5 the checkpoint to resume execution in the normal execution mode from the launch  
6 point instruction (the instruction that originally encountered the launch point stall  
7 condition).

1           15. The apparatus of claim 10, wherein while executing deferred  
2 instructions in the deferred mode, the execution mechanism is configured to:  
3           issue deferred instructions for execution in program order;  
4           defer execution of deferred instructions that still cannot be executed  
5 because of unresolved data dependencies; and to  
6           execute other deferred instructions that are able to be executed in program  
7 order.

1           16. The apparatus of claim 15, wherein if all deferred instructions are  
2 executed in the deferred mode, the execution mechanism is configured to return to  
3 a normal execution mode to resume normal program execution from the point  
4 where the execute-ahead mode left off.

1           17. The apparatus of claim 10, wherein the unresolved data  
2 dependency can include:  
3           a use of an operand that has not returned from a preceding load miss;  
4           a use of an operand that has not returned from a preceding translation  
5 lookaside buffer (TLB) miss;  
6           a use of an operand that has not returned from a preceding full or partial  
7 read-after-write (RAW) from store buffer operation; and  
8           a use of an operand that depends on another operand that is subject to an  
9 unresolved data dependency.

1           18. The apparatus of claim 10, wherein the non-data-dependent stall  
2 condition can include:  
3           a memory barrier operation;  
4           a load buffer full condition; and

5 a store buffer full condition.

1 19. A computer system that dynamically adjusts the aggressiveness of  
2 an execute-ahead processor, comprising:

3 an execute-ahead processor;

4 a memory;

5 an execution mechanism within the execute-ahead processor configured to  
6 execute instructions in an execute-ahead mode, wherein instructions that cannot  
7 be executed because of an unresolved data dependency are deferred, and other  
8 non-deferred instructions are executed in program order, and wherein if a non-  
9 data-dependent stall condition is encountered, the execution mechanism is  
10 configured to enter a scout mode, wherein instructions are speculatively executed  
11 to prefetch future loads, but results are not committed to the architectural state of  
12 the execute-ahead processor;

13 wherein if an unresolved data dependency is resolved during the execute-  
14 ahead mode, the execution mechanism is configured to execute deferred  
15 instructions in a deferred mode;

16 wherein if some instructions are deferred again during the deferred mode,  
17 the execution mechanism is configured to,

18 determine whether to resume execution in the execute-  
19 ahead mode,

20 if it is determined to do so, to resume execution in the  
21 execute-ahead mode, and

22 otherwise to resume execution in a non-aggressive mode.

1 20. The computer system of claim 19, wherein while resuming  
2 execution in the non-aggressive execution mode, the execution mechanism is

3       configured to remain in the deferred mode until all deferred instructions are  
4       executed and the execution mechanism returns to a normal execution mode.

1           21.      The computer system of claim 19, wherein while resuming  
2       execution in the non-aggressive execution mode, the execution mechanism is  
3       configured to resume execution in a non-aggressive execute-ahead mode, wherein  
4       if a non-data-dependent stall condition is encountered, the execution mechanism  
5       does not enter the scout mode, but instead waits for the non-data-dependent stall  
6       condition to be resolved, or for an unresolved data dependency to return, before  
7       proceeding.